

# OpenBDLM, an Open-Source Software for Structural Health Monitoring using Bayesian Dynamic Linear Models

Ianis Gaudot

*Postdoc, Dept. of Civil, Geological and Mining Engineering, Polytechnique Montreal*

Luong Ha Nguyen

*Ph.D. student, Dept. of Civil, Geological and Mining Engineering, Polytechnique Montreal*

Shervin Khazaeli

*Ph.D. student, Dept. of Civil, Geological and Mining Engineering, Polytechnique Montreal*

James-A. Goulet

*Assistant professor, Dept. of Civil, Geological and Mining Engineering, Polytechnique Montreal*

**ABSTRACT:** During the last decade, the rise of sensing technologies fostered the development of new data-driven Structural Health Monitoring (SHM) techniques. Among them, Bayesian Dynamic Linear Models (BDLMs) are capable of isolating the baseline responses of civil infrastructure from external effects, thus allowing to interpret the intrinsic behavior of civil structure and to detect anomalies. The generalization of BDLMs for SHM borrows tools from many fields, and there is currently no standalone software allowing BDLMs to be used routinely by practitioners. This study intends to bridge this gap by introducing OpenBDLM, a Matlab open-source software specifically developed to use BDLMs for SHM. In this paper, synthetic dataset is examined to illustrate the functionalities of the software, from data pre-processing to results visualization.

During the last decade, the rise of sensing technologies fostered the development of new data-driven structural health monitoring (SHM) techniques (Farrar and Worden, 2012). One challenge in SHM is to separate the baseline response of a structure (that carries irreversible change due to the ageing of the structure) from the reversible changes in the behavior due to external effects (temperature, traffic, etc.). The extraction of the baseline is crucial for the detection of changes in the baseline response (i.e. anomalies). In many cases, external effects variation might be greater than the true damaged-induced variation, thus leading to

false anomaly detection. One approach is to decompose the observed structural responses into a set of components, among them the baseline response of the structure. Multiple linear regression (Gamse and Oberguggenberger, 2017) and neural network (Mata, 2011) were found to be effective techniques for achieving such a purpose. However, without retraining, they are not capable to self-adaptation due to changes in structural/environmental conditions (i.e. non-stationarity). This is a key limitation because an efficient and autonomous anomaly detection tool requires analyzing non-stationary time-series online. Bayesian Dynamic Linear Model BDLM, is a class of state-

space model that allows non-stationary components to be learned online (West and Harrison, 1999). It is interesting to note that BDLM have been mainly used in econometrics so far. One possible reason is that several open-source softwares for macroeconomics and finance analysis based on state-space models have been made available since the 80's (see Commandeur et al., 2011, for a review). In contrast, only few studies have explored the use of BDLM for SHM (Solhjell, 2009; Goulet, 2017). Recently, Goulet and Koo (2018); Nguyen and Goulet (2018b) and Nguyen and Goulet (2018a) have shown that BDLM can be used to track time-varying baseline structure response and detect anomalies. The generalization of BDLM for SHM borrows tools from many disciplines, including data processing, applied statistics, and machine learning. There is currently no standalone software allowing BDLMs to be used routinely by SHM practitioners. This study intends to bridge this gap by introducing OpenBDLM, a software specifically developed to use BDLM for SHM. OpenBDLM is written in Matlab to be fully cross platform. The software is interactive and documented to facilitate its use. It is open source and free with the aim of spreading out the use of BDLMs for SHM, as well as building a community of developers to further improve the code. This paper is organized as follows. The first section briefly reviews the BDLM theory. The second section summarizes the main software functionalities. The third section presents a case-study based on synthetic data to show how the software is used in practice, and finally, the conclusion addresses some perspectives for further improvements of the software.

## 1. BAYESIAN DYNAMIC LINEAR MODELS

### 1.1. Linear gaussian state-space model

Bayesian dynamic linear models are a class of linear gaussian state-space models which can be described from the transition and the observation equations (West and Harrison, 1999). The transition equation describes the dynamics of the system, and is formulated as

$$\mathbf{x}_t = \mathbf{A}_t \mathbf{x}_{t-1} + \mathbf{w}_t, \quad \begin{cases} \mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \\ \mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t), \end{cases} \quad (1)$$

where, for each each time  $t = 1, \dots, T$ , the variables  $\mathbf{x}_t$  follow a Gaussian distribution with mean  $\boldsymbol{\mu}_t$  and covariance matrix  $\boldsymbol{\Sigma}_t$ ,  $\mathbf{A}_t$  is the transition matrix, and  $\mathbf{w}_t$  represents Gaussian model errors with zero mean and covariance matrix  $\mathbf{Q}_t$ . The variables  $\mathbf{x}_t$  are usually referred to as hidden states because they are not directly observed. The relationship between the observations  $\mathbf{y}_t$  and the hidden states  $\mathbf{x}_t$  is given by the observation equation, such as

$$\mathbf{y}_t = \mathbf{C}_t \mathbf{x}_t + \mathbf{v}_t, \quad \left\{ \begin{array}{l} \mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_t), \end{array} \right. \quad (2)$$

where  $\mathbf{C}_t$  is the observation matrix, and  $\mathbf{v}_t$  is the Gaussian measurement error with zero mean and covariance matrix  $\mathbf{R}_t$ . BDLMs are capable of analyzing multiple time series simultaneously. In case of dependencies between multiple time series, regression coefficients are added in  $\mathbf{C}_t$  (Goulet, 2017). One particularity of BDLMs is their capacity to update the current estimated state with the current observations, thus allowing to perform online state inference of non-stationary time series.

### 1.2. Kalman filter

The analytical solutions for the prediction, observation and update step are available through the Kalman filter (KF), which can be expressed in its short form as:

$$(\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}, \mathcal{L}_t) = \text{Filter}(\boldsymbol{\mu}_{t-1|t-1}, \boldsymbol{\Sigma}_{t-1|t-1}, \mathbf{y}_t, \mathbf{A}_t, \mathbf{Q}_t, \mathbf{C}_t, \mathbf{R}_t) \quad (3)$$

where  $\mathcal{L}_t$  is the marginal likelihood describing the probability of observing observations  $\mathbf{y}_t$  at time  $t$  given all the observations up to time  $t - 1$ . The standard Kalman filter expressed in Eq. 3 can process stationary, trend stationary, and acceleration stationary time series, but it is not capable of handling non-stationary time-series, which is a major limitation when it comes to anomaly detection. The generalization of the Kalman Filter for non-stationary time-series is found in the Switching Kalman filter (SKF) equations.

### 1.3. Switching Kalman filter

In the context of SHM, we are interested in anomaly detection, that is, modelling and detecting the changes of behavior due to changes in the dynamics of the baseline response of the structure. One

way to model changing dynamics is to run in parallel a collection of  $S$  linear models, each having their own system dynamics  $\mathbf{A}_t$  and  $\mathbf{Q}_t$ . In such approach, a discrete markovian switching variable  $s_t = 1, \dots, j, \dots, S$  with a transition probabilities matrix  $\mathbf{Z}_t$  and probabilities  $\pi_t$  is introduced to indicate which dynamics is used at time  $t$ . The problem of incorporating switching dynamics into the model is that the state vector grows in a way that the dimension of the state vector at time  $t$  is  $S^t$ . Therefore, the estimation quickly becomes intractable. One solution is to merge at each time  $t$  the states sharing the same dynamics using gaussian mixture. This technique, known as the Switching Kalman filter (SKF, Murphy, 1998), allows to keep the dimension of the state vector equal to  $S$  at each time  $t$ . The SKF algorithm can be divided into two successive steps, (i) the “Filter” step and, (ii) the “Collapse” step. Following the notation used in Eq. 3 the first step can be expressed in its short form as:

$$(\boldsymbol{\mu}_{t|t}^{i(j)}, \boldsymbol{\Sigma}_{t|t}^{i(j)}, \mathcal{L}_t^{i(j)}) = \text{Filter}(\boldsymbol{\mu}_{t-1|t-1}^i, \boldsymbol{\Sigma}_{t-1|t-1}^i, \mathbf{y}_t, \mathbf{A}_t^j, \mathbf{Q}_t^{i(j)}, \mathbf{C}_t^j, \mathbf{R}_t^j) \quad (4)$$

where the superscripts  $i(j)$  indicates that the current state at time  $t$  is  $s_t = j$  given the state at time  $t - 1$  is  $s_{t-1} = i$ , and  $\mathcal{L}_t^{i(j)}$  the marginal likelihood that describes the probability of observing observations  $\mathbf{y}_t$  at time  $t$  given all the observations up to time  $t - 1$ , and given the state at time  $t - 1$  was  $s_{t-1} = i$  and that it switches to  $s_t = j$  at time  $t$ . The state probability  $\pi_{t|t}^j$  at each time  $t$  is computed from the previous state probabilities  $\pi_{t-1|t-1}^i$ , the likelihood  $\mathcal{L}_t^{i(j)}$ , and the transition probability  $Z_t^{i(j)}$ , such as

$$\pi_{t|t}^j = \sum_{i=1}^S \frac{\mathcal{L}_t^{i(j)} \pi_{t-1|t-1}^i Z_t^{i(j)}}{c}, \quad (5)$$

where  $c$  is a normalization constant ensuring that  $\sum_{j=1}^S \pi_{t|t}^j = 1$ . Moreover, the state switching probability is defined as

$$W_{t-1|t}^{i(j)} = \frac{\mathcal{L}_t^{i(j)} \pi_{t-1|t-1}^i Z_t^{i(j)}}{c \pi_{t|t}^j}. \quad (6)$$

$W_{t-1|t}^{i(j)}$  are required to perform the “Collapse” step, which can be expressed in its short form as:

$$(\boldsymbol{\mu}_{t|t}^j, \boldsymbol{\Sigma}_{t|t}^j) = \text{Collapse}(\boldsymbol{\mu}_{t|t}^{i(j)}, \boldsymbol{\Sigma}_{t|t}^{i(j)}, W_{t-1|t}^{i(j)}); \quad (7)$$

where state switching probabilities  $W_{t-1|t}^{i(j)}$  are used as weighting factors for the gaussian mixture. From Eq. 7, it is clear that the SKF algorithm provides a set a  $S$  state vectors at each time  $t$ . However, for the ease of interpretation, it is generally more convenient to have a single state vector at each time  $t$ . Therefore, we hereafter introduce the “Merge” step. Similarly to the “Collapse” step of the SKF algorithm, the “Merge” step uses the gaussian mixture technique, and it can be expressed in its short form as:

$$(\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}) = \text{Merge}(\boldsymbol{\mu}_{t|t}^j, \boldsymbol{\Sigma}_{t|t}^j, \pi_{t|t}^j); \quad (8)$$

where the state probabilities  $\pi_{t|t}^j$  is used as weighting factors for the gaussian mixture.

#### 1.4. Maximum Likelihood Estimation

The matrices  $\mathbf{A}_t$ ,  $\mathbf{Q}_t$ ,  $\mathbf{C}_t$  and  $\mathbf{R}_t$  depend on a set of static model parameters  $\boldsymbol{\theta}$ . In most cases,  $\boldsymbol{\theta}$  are unknown, and they can be learned from a training dataset  $\mathbf{y}_{1:\text{Tr}}$ . The Maximum (log-) Likelihood Estimation (MLE) is a technique often employed to learn from the data the optimal set of model parameters, noted  $\boldsymbol{\theta}^*$ , such as

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} [\mathcal{L}_{\text{Tr}}(\boldsymbol{\theta})], \quad (9)$$

where  $\mathcal{L}_{\text{Tr}}(\boldsymbol{\theta})$  is the marginal log-likelihood defined as,

$$\mathcal{L}_{\text{Tr}}(\boldsymbol{\theta}) = \sum_{t=1}^{\text{Tr}} \ln \left[ \sum_{j=1}^S \sum_{i=1}^S \mathcal{L}_t^{i(j)} \pi_{t-1|t-1}^i Z_t^{i(j)} \right], \quad (10)$$

where implicit dependency on  $\boldsymbol{\theta}$  is considered in the right side of the equation to simplify the notation. Note that  $\mathcal{L}_{\text{Tr}}(\boldsymbol{\theta})$  is directly computed from the Switching Kalman filter, which provides the values of  $\mathcal{L}_t^{i(j)}$  and  $\pi_{t-1|t-1}^i$  (see Eq. 4 and Eq. 5); the values of  $Z_t^{i(j)}$  are known from the current set of model parameters.

## 2. SOFTWARE FUNCTIONALITIES

In this section, the software functionalities are described. The global software workflow is summarized in Figure 1.

### 2.1. Data pre-processing

As mentioned in Section 1.1, BDLMs are capable to analyze simultaneously multiple time-series. Multiple time-series analysis is useful to incorporate information from observed environmental effects by creating dependencies between several time-series. However, in most cases, the set of available time-series is heterogeneous, in the sense that each time-series does not originate from the same system of acquisition. Therefore, the raw data do not usually share the same timestamps. This is an issue because BDLM techniques are not capable to analyze asynchronous time-series.

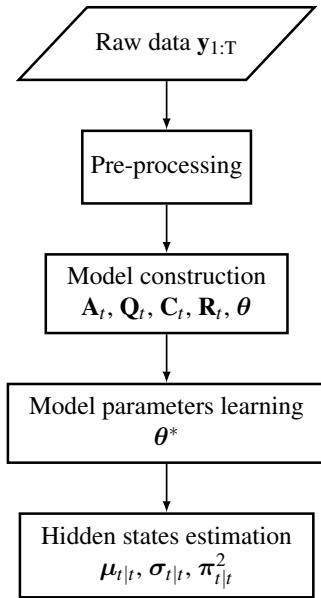


Figure 1: OpenBDLM workflow

The data pre-processing in OpenBDLM is thus dedicated to synchronize the time-series between each others. By default, the time synchronization is done by replacing the corresponding missing values in the time series with missing data. Custom pre-processing is also possible to control the final amount of missing data in the dataset, or to perform resampling by data averaging over time-windows of fixed length. Moreover, the data pre-processing in OpenBDLM is also used to choose the time-series

to process, and to select the period of analysis. The time synchronization is updated automatically as time-series are added to (or removed from) the dataset. The preprocessing in OpenBDLM is minimalist and focuses on time synchronization. Neither outlier removal nor normalization is done to preserve the genuine information from the data.

### 2.2. Model construction

BDLMs are used to decompose time-series into a set of hidden state variables. The choice of the type of components associated with each time-series is part of the model construction. Each sub-component has its own model and parameter vector (i.e. a set of  $A_t$ ,  $Q_t$ ,  $C_t$ , and  $R_t$  matrices) which are then assembled to form the full model. OpenBDLM supports three types of components: (1) baseline, (2) periodic, and (3) autoregressive. (1) The baseline component models the local mean of the time series. For the time series directly related to the structural behavior (displacement, frequency), the baseline component is of main interest because it carries the baseline response of the structure. There are three types of baseline proposed in the software: (i) level only model, (ii) trend model, (iii) acceleration model. (2) The periodic component models harmonic periodic phenomena, which are most often related to external effects (temperature for instance). (3) The autoregressive component models the time dependent model error. Note that each component can be replicated, each having its own set of model parameters. For instance, two periodic components with periods of 365 days and 1 day can be used to model seasonal and daily variations, respectively. OpenBDLM enables to build complex models easily, while automatically assigning default values for the model parameters in the corresponding matrices.

### 2.3. Model parameters learning

The default values for the model parameters assigned during model construction are typically poor guesses estimated from heuristic knowledge, and a better estimation must be learned from the data using the MLE procedure presented in Section 1.4. OpenBDLM implements the Newton-Raphson (NR, Gelman et al., 2004) and Stochas-

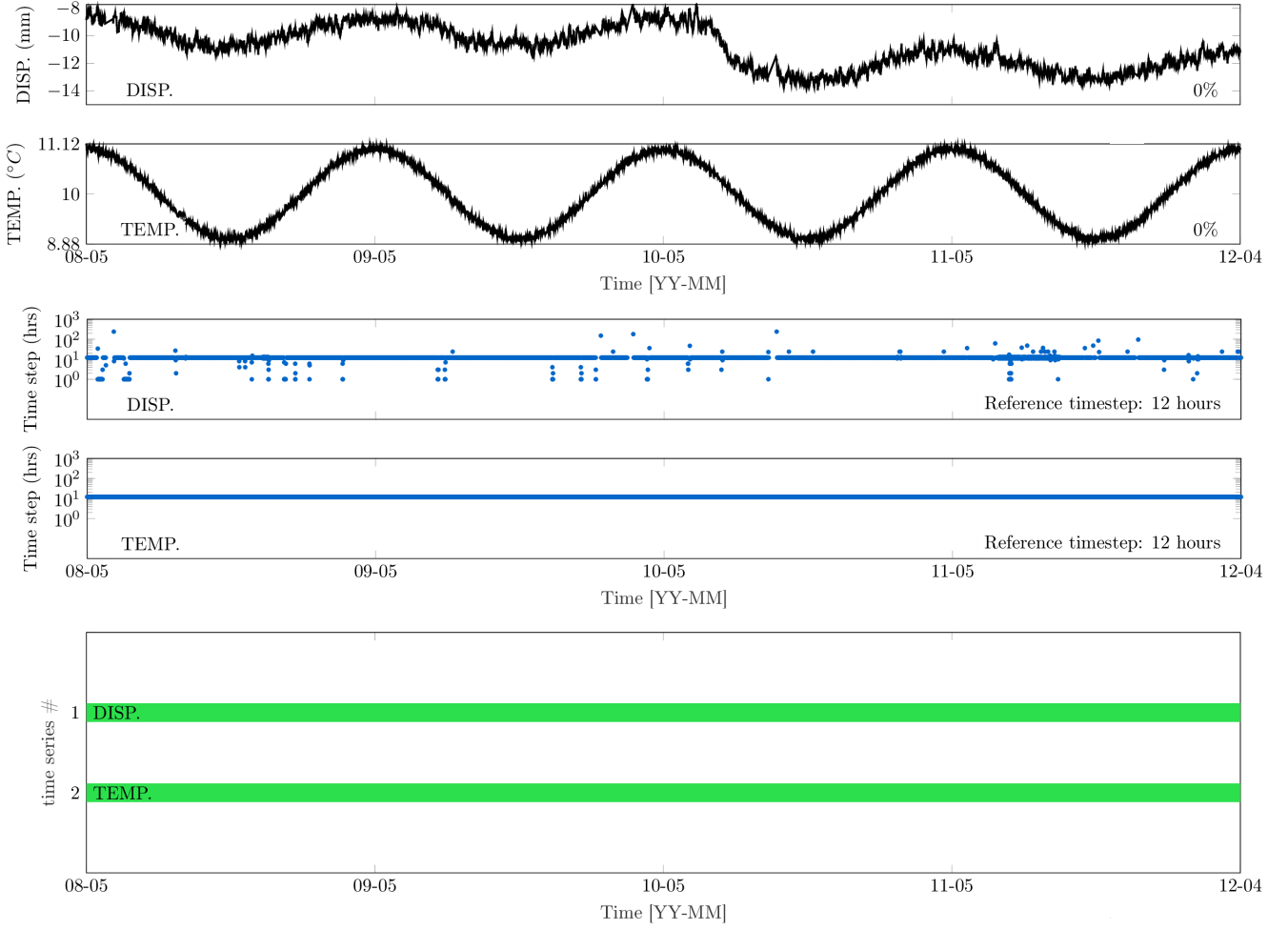


Figure 2: Simulated displacement (DISP.) and temperature (TEMP.) time series. From top to bottom, the graphs represent the amplitude, the time steps and the working period as well as the position of the missing data in each time series, respectively. (This figure and the following ones are graphical outputs of OpenBDLM).

tic Gradient Descent (SGD, Goodfellow et al., 2016) iterative gradient-based optimization techniques to achieve this task. The NR and SGD algorithms involve the computations of the first and second order derivatives of the log-likelihood. The derivatives are approximated using the finite-difference method, and OpenBDLM takes advantages of the CPU parallel computing (when available) to speedup the computations. Defaults values for the starting model parameters, learning rate, stopping criterion, and number of iterations are provided by OpenBDLM, but custom setting is possible. Note that the NR and SGD algorithm are likely to be trapped in a local maxima if the starting initial values of the models parameters are far from the true values of parameters. Therefore, it is advised

to repeat the algorithm several times with different initial starting model parameters values to evaluate the robustness of the results.

#### 2.4. Hidden states estimation

The mean  $\mu_{t|t}$  and the standard deviation  $\sigma_{t|t} = \sqrt{\text{diag}(\Sigma_{t|t})}$  of each estimated hidden component are computed at each time  $t$  using the Switching Kalman filter algorithm and the “Merge” step (see Eq. 4-8). OpenBDLM creates one figure for each component. The plots can be exported in PNG, PDF and  $\text{\LaTeX}$  format for publication and reporting.

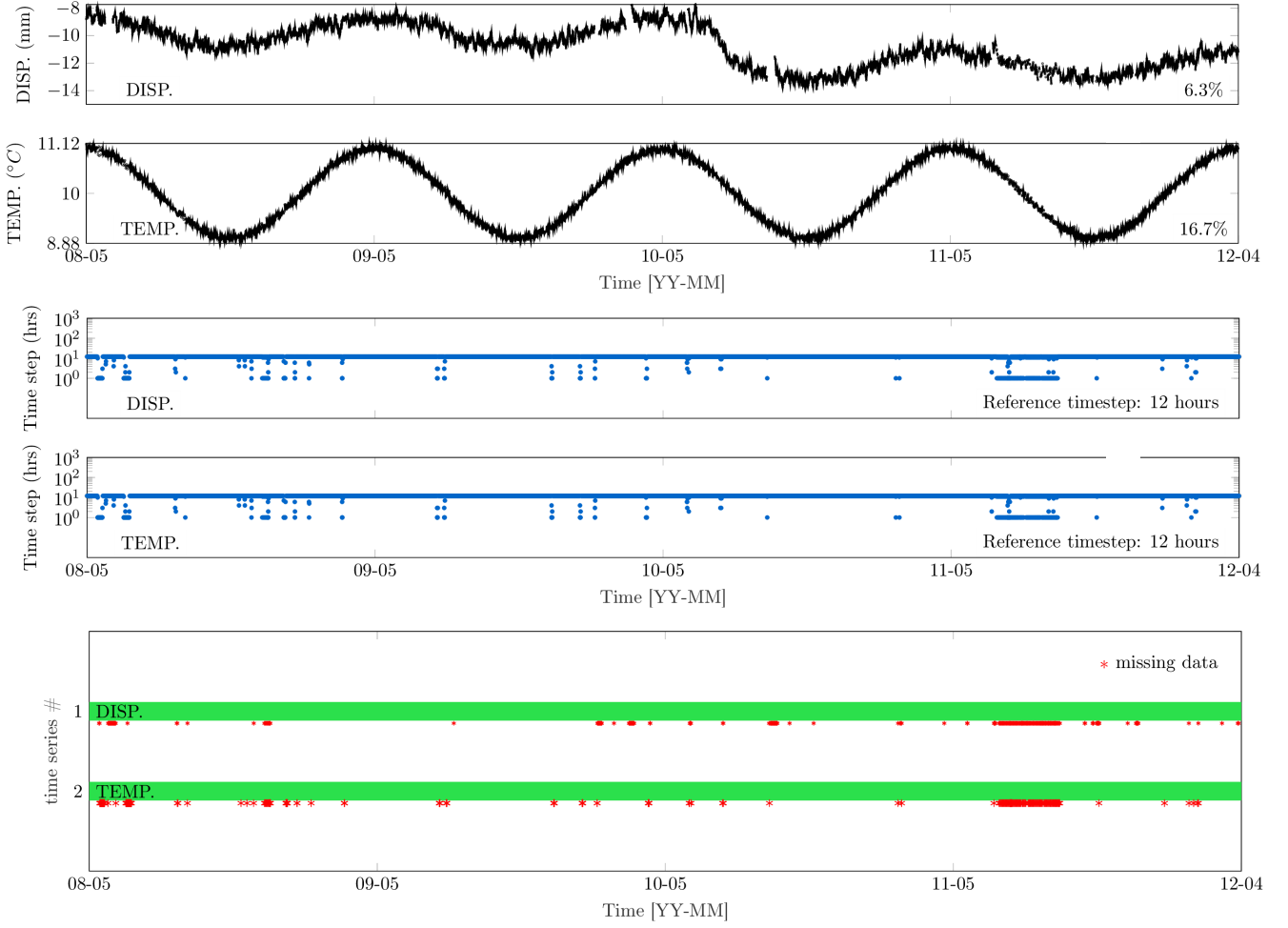


Figure 3: Effects of the data pre-processing. Same legend as Figure 2. In the bottom graph, the crosses indicate missing data.

### 2.5. Anomaly detection

The Switching Kalman filter presented in Section 1.3 enables to run multiple model dynamics in parallel, and to compute the probability of each model  $\pi_{t|t}$  at each time  $t$ . In OpenBDLM, a two-models Switching Kalman filter (i.e.  $S = 2$ ) can be implemented to detect anomalies. The first and second model dynamics are assumed to be related to the normal and abnormal behavior expected for a given structure, respectively. The detection of an anomaly occurs when the probability of the second model is close to 1. OpenBDLM aims at detecting changes of behavior due to changes in dynamics in the baseline response of the structure. Therefore, the software handles model switching between the three types of baseline dynamics described in Section 2.2, that is, local level, local trend, and lo-

cal acceleration models. Considering a maximum of two model dynamics, the software supports six types of model switch: (1) from local level model to local trend model (and reverse), (2) from local level model to acceleration model (and reverse), (3) from local trend to acceleration model (and reverse). The model probability of the abnormal model  $\pi_{t|t}^2$  is plotted at each time  $t$  to visualize the presence of anomalies in the time-series.

## 3. CASE STUDY

The software functionalities are illustrated on a case-study that involves a simulated displacement and a simulated temperature time-series (labelled DISP. and TEMP. respectively, see Fig. 2, top). Both time series span a period of 4 years between May 2008 and April 2012. This simulated dataset

```

1 %% A — Project name
2 misc.ProjectName='DISPTMP_ICASP13';
3
4 %% B — Data
5 dat=load('DATA_DISPTMP_ICASP13.mat');
6 data.values=dat.values;
7 data.timestamps=dat.timestamps;
8 data.labels={'DISP.', 'TEMP.'};
9
10 %% C — Model structure
11 % Model components
12 model.components.block{1}=[23 41 ] ...
13     [21 31]]; % Model 1
14 model.components.block{2}=[13 41 ] ...
15     [21 31]]; % Model 2
16
17 % Model component constrains
18 model.components.const{2}=[0 1 ] [1 1 ] };
19
20 % Model inter-components dependence
21 model.components.ic={2 } [ ] };

```

Listing 1: OpenBDLM configuration file for the case-study presented in Section 3

mimics the situation in which the structural response (DISP.) depends on an observed external effect (TEMP.). On June 26, 2010, a switch from stationary to trend stationary has been added to mimic a fictitious anomaly in the DISP. baseline. The anomaly lasts 30 days. The timestep vector for the DISP. time series is non-uniform, with timesteps varying from 1 to 72 hours (Fig. 2, middle). On the other hand, the timestep vector for TEMP. time-series is uniform with a fixed value of 12 hours. In such case, pre-processing is required to synchronize the time series between each others before performing a BDLM analysis (see Section 2.1). Therefore, OpenBDLM adds missing data to synchronize the time series between each others (Fig. 3, bottom). The dataset is now ready for BDLMs analysis because the timesteps are identical for each time-series (Fig. 3, middle). Here, the goal is to extract the baseline response of the DISP. time-series. Thus, a two-models SKF is configured, in which the first model includes a trend baseline component, and the second model includes an accelera-

tion baseline component. The periodic patterns observed in the displacement is modelled by creating a dependency on the observed temperature time series. OpenBDLM uses a configuration file for initializing a project, loading the data, and building the model (see Listing 1). Note that less than 22 lines of code is necessary to initialize a project. Each model component is associated with a reference number for compact notation and readability (local trend model = 13, periodic model = 31, autoregressive = 41, etc.). OpenBDLM also proposes to build the configuration file from command line user's interaction.

The model parameters are learned from the full dataset using the Newton-Rapshon algorithm presented in Section 2.3. The mean and standard deviation of each hidden component are then estimated using the SKF algorithm followed by the merging step, presented in Section 1.3. After the analysis, OpenBDLM creates a set of plots to visualize and interpret the results. Among them, the mean ( $\mu_{t|t}^B$ ) and standard deviation ( $\sigma_{t|t}^B$ ) of the baseline component extracted for the displacement time series as well as the probability of the “abnormal” model ( $\pi_{t|t}^2$ ), which are presented in Fig. 4. The estimated baseline is close to the true values represented by the dashed lines. After a relatively short delay of around 17 days, the model catches the change in the structure baseline response (Fig. 4, top). This change in the baseline dynamics is seen in the model probability plot (Fig. 4, bottom). Indeed, the probability of the second model reaches one on July 17, 2010, resulting in an anomaly detection 20 days later than the true onset of the anomaly.

#### 4. CONCLUSIONS

The paper introduces OpenBDLM, a new open-source Matlab software to perform Bayesian dynamic linear modeling for Structural Health Monitoring. The software package can be downloaded from GitHub. OpenBDLM is being developed for Structural Health Monitoring, but it is well suited to process any time series with time steps of the order of one hour or higher.



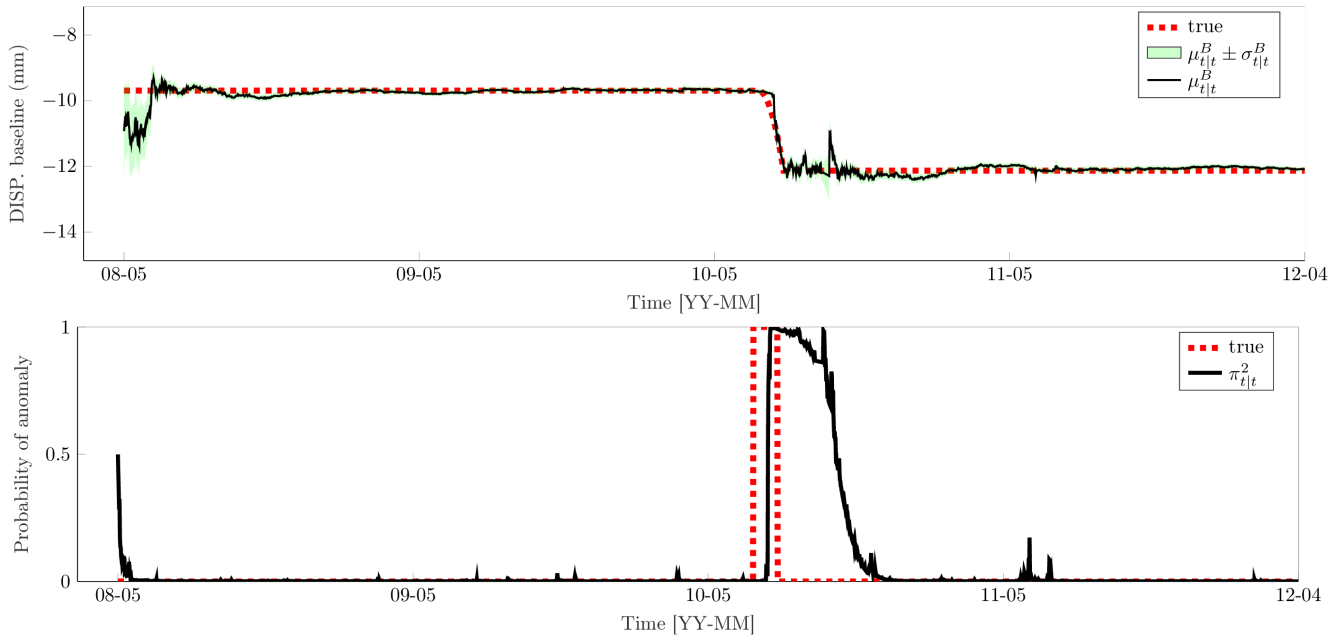


Figure 4: Switching Kalman filter results. Top: Estimated baseline component for the simulated displacement time series. Bottom: probability of the model 2 (trend model). The dashed lines indicate the true values.

## 5. REFERENCES

- Commandeur, J., Koopman, S., and Ooms, M. (2011). “Statistical software for state space methods.” *Journal of Statistical Software, Articles*, 41(1), 1–18.
- Farrar, C. and Worden, K. (2012). *Structural Health Monitoring: A Machine Learning Perspective*. Wiley.
- Gamse, S. and Oberguggenberger, M. (2017). “Assessment of long-term coordinate time series using hydrostatic-season-time model for rock-fill embankment dam.” *Structural Control and Health Monitoring*, 24(1).
- Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd ed. edition.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press <http://www.deeplearningbook.org>.
- Goulet, J. (2017). “Bayesian dynamic linear models for structural health monitoring.” *Structural Control and Health Monitoring*, 24.
- Goulet, J. and Koo, K. (2018). “Empirical validation of bayesian dynamic linear models in the context of structural health monitoring.” *Journal of Bridge Monitoring*, 23.
- Mata, J. (2011). “Interpretation of concrete dam behaviour with artificial neural network and multiple linear regression models.” *Engineering Structures*, 33(3), 903 – 910.
- Murphy, K. P. (1998). “Switching kalman filters.” *Report no.*, Citeseer.
- Nguyen, L. and Goulet, J.-A. (2018a). “Anomaly detection with the switching kalman filter for structural health monitoring.” *Structural Control and Health Monitoring*, 25.
- Nguyen, L. and Goulet, J.-A. (2018b). “Structural health monitoring with dependence on hidden non-harmonic covariates.” *Engineering Structures*, 166, 187–194.
- Solhjell, I. (2009). “Bayesian forecasting and dynamic models applied to strain data from the göta river bridge.” M.S. thesis, University of Oslo,
- West, M. and Harrison, J. (1999). *Bayesian Forecasting and Dynamic Models*. Springer Series in Statistics. Springer New York.